DOCUMENT RESUME

ED 311 991 JC 890 498

TITLE Software Snapshots: Where Are You in the Picture? An Invitation to Community College Faculty.

INSTITUTION Interuniversity Communications Council (EDUCOM), Princeton, N. J.; Miami-Dade Community Coll. District, FL.

PUB DATE 89

NOTE 25p.; Illustrations may not reproduce clearly.

AVAILABLE FROM League for Innovation in the Community College, 25431 Cabot Road #204, Laguna Hills, CA 92653.

PUB TYPE Guides - Non-Classroom Use (055)

EDRS PRICE MF01/PC01 Plus Postage.

DESCRIPTORS Community Colleges; *Computer Software; *Courseware; Learning Modules; *Material Development; *Teacher Developed Materials; *Teacher Role; Two Year Colleges

ABSTRACT

Designed to encourage community college faculty to examine the opportunities and responsibilities for educational software development and use, this booklet offers an overview of the steps involved in developing quality software. After reviewing some of the innovative uses of software in education, the booklet focuses on the readers' goals for their own teaching and for their students' learning to help them assess their readiness to integrate teaching and technology. In the next section, guidelines are provided for searching for, selecting, using, and adapting existing educational software. After urging instructors to start with a simple module, the booklet outlines the six stages in software development: (1) analysis of the objectives of the project, the approach, and the hardware/software requirements; (2) program design; (3) programming; (4) preparation of documentation, including a student guide and instructor's manual; (5) evaluation and debugging; and (6) maintenance. Alternatives to working alone on development projects are discussed next, including serial, collaborative, and parallel teams. After comparing the use of general purpose languages, authoring languages, and authoring systems, the booklet warns of the time commitment needed for serious software development. Issues to be considered before embarking on a software development project are also addressed, including institutional priorities, hardware/software choices, and the software marketplace. Finally, the intrinsic and extrinsic benefits of software development are reviewed. A list of associations, non-commercial software distributors, journals, and bulletin boards is attached. (AYC)

****
* Reproductions supplied by EDRS are the best that can be made *
* from the original document. *
****

SOFTWARE SNAPSHOTS

ED311991

"PERMISSION TO REPRODUCE THIS MATERIAL HAS BEEN GRANTED BY
Kamala Anandam
TO THE EDUCATIONAL RESOURCES INFORMATION CENTER (ERIC)."

U.S. DEPARTMENT OF EDUCATION
Office of Educational Research and Improvement
EDUCATIONAL RESOURCES INFORMATION CENTER (ERIC)
X This document has been reproduced as received from the person or organization originating it.
☐ Minor changes have been made to improve reproduction quality.
• Points of view or opinions stated in this document do not necessarily represent official OERI position or policy.

JC 890 498

Where Are You In The Picture?

AN INVITATION TO COMMUNITY COLLEGE FACULTY

ACKNOWLEDGEMENTS

The EDUCOM Software Initiative is grateful to:

Miami-Dade Community College, the Pew Charitable Trusts, and The Software Publishers Association for their direct or indirect support of this publication,

IBM/ACIS for funding the distribution of this publication,

The League for Innovation in the Community College for distributing this publication,

and

EDUCOM Corporate Leadership Sponsors
Apple Computer, Inc.
IBM Corporation.

The use in this directory of registered trade names, trademarks, etc. without special acknowledgement does not imply that such names, as defined by relevant protection laws, may be regarded as unprotected and, thus, free for general use.

SOFTWARE SNAPSHOTS:
WHERE ARE YOU IN THE PICTURE?

by

AN EDUCOM SOFTWARE INITIATIVE PROJECT COMMITTEE

Kamala Anandam (Chair)
IBM/ACIS Consulting Scholar
Associate Dean
Division of Educational Technologies
Miami-Dade Community College
Miami, FL

Robert Cavalier
Assistant Director of ICEC
Center for Design of Educational Computing
Carnegie-Mellon University of Pittsburgh, PA

Carol Dwyer
Instructional Designer
Computer-Based Education Laboratory
Penn State University
University Park, PA

John J. Hirschbuhl
Director
Center for Computer-Based Education
University of Akron
Akron, OH

Norm Nicholson
Director
Computer Services
Cerritos College
Norwalk, CA

George Voegel
Dean
Educational Services
Harper College
Palatine, IL

Edward Zoll
Professor
Mathematics/Computer Science
Kean College of New Jersey
Union, NJ

Reviewed by

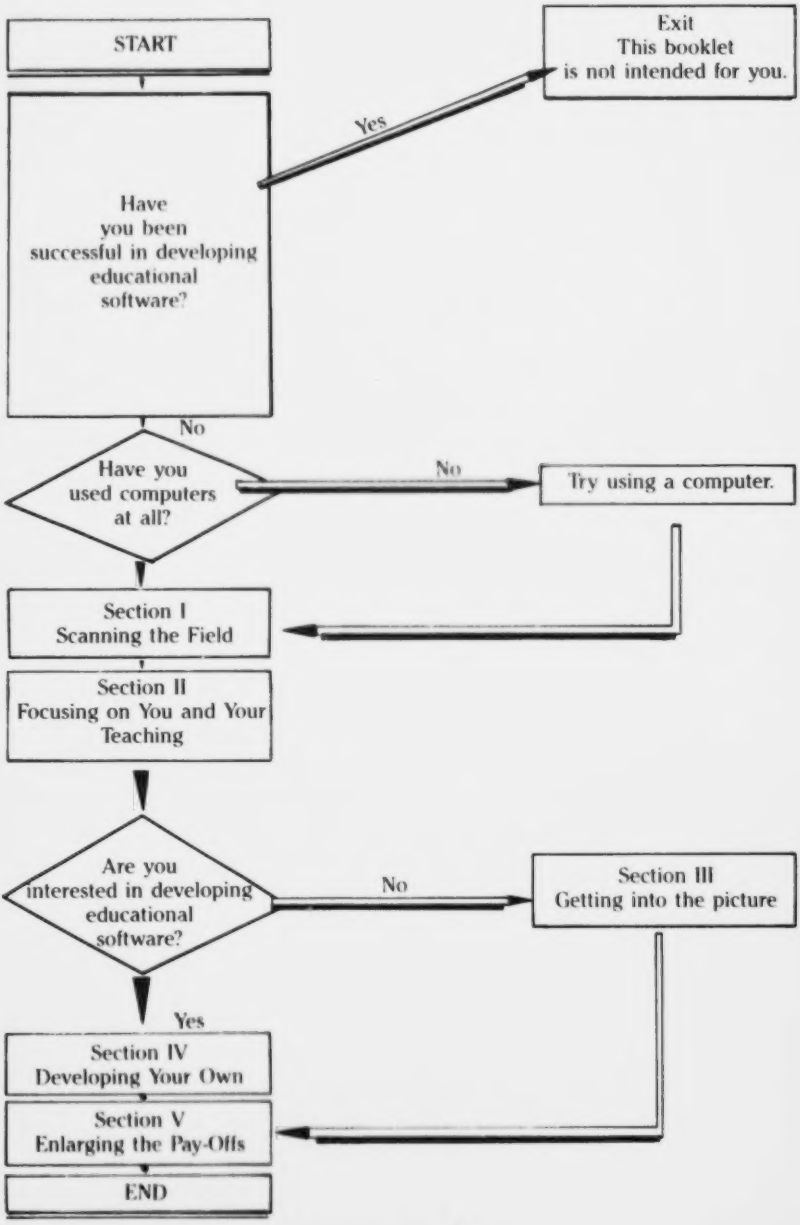
Rudy Gentry
Director
Computer and Inf. Systems
Johnson County Community College
Overland Park, KS

Michael Hoban
Developmental Studies
LaGuardia Community College
The City University of N.Y.
Long Island City, NY

Celeste Schwartz
Chief Information Officer
Montgomery County Community College
Bluebell, PA

Tim Songer
Coordinator
Project READY
Central Piedmont Community College
Charlotte, NC

TABLE OF CONTENTS



SCANNING THE FIELD

WHY THIS BOOK? The time is right for you to consider more seriously the use and development of educational software for several reasons. Computers continue to improve in quality, cost, and accessibility; software-authoring tools continue to improve in their effectiveness, ease of use, and transportability; and quality educational software packages continue to become increasingly available for faculty to use or adapt. When you use or develop educational software, you have an opportunity to contribute to the educational community in two ways—by enhancing the teaching/learning process and by increasing the availability of tested, good-quality educational software. Though quality software use or development is not for the faint of heart, we believe that many faculty such as you should consider the integration of teaching and technology. This book is not, however, a "how-to" manual. It is merely an invitation, albeit a strong one, for you to examine the opportunities and responsibilities of educational software development and/or use.

EDUCATIONAL SOFTWARE IN HIGHER EDUCATION The ideas of faculty and the sophistication of technology are converging rapidly to produce some meaningful and exciting learning opportunities for students.

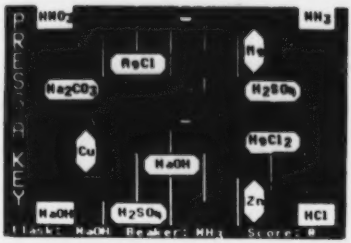
Consider the following examples of software being used in educational institutions. Some faculty have developed the software, others have used it, and some have creatively adapted it to meet their own special needs.

The examples that follow embody a few of the ideas that some faculty members brought to the technological arena. By and large, the stimulus for their ideas was their interest in improving their teaching and in enhancing their students' learning. What are your ideas?

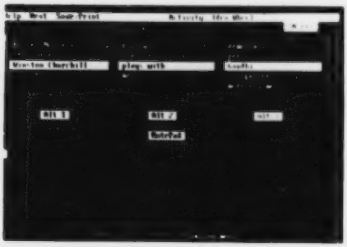
Students simulate a series of muscle contraction experiments and observe that efficient exertion wins the race in **Mechanical Properties of Active Muscle.***

Students become familiar with the **Scientific Method** of inquiry by working their way through the "switched label" problem.*

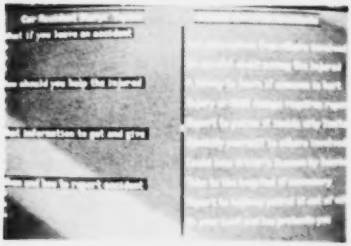
Students do real-time collection of observational data about social behavior and interaction in **Eventlog**, a psychology tool.*



Students master the periodic table, chemical notation and chemical reactions in a chemistry tutorial, **Introduction to Chemistry.***



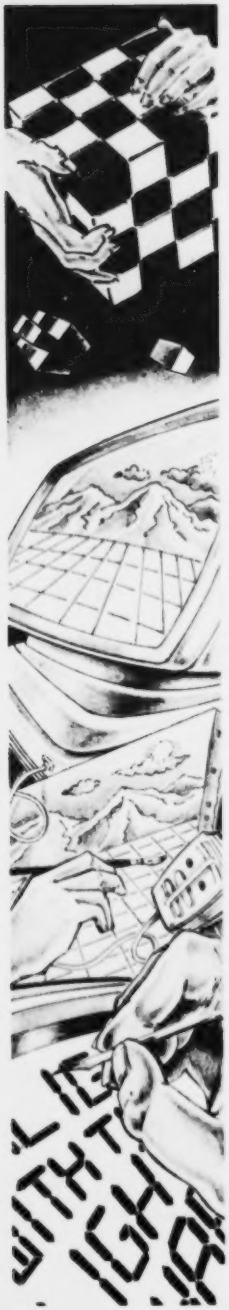
Students use the Idea Wheel in **Writer's Helper** to generate ideas for their essay.*



Students match details from a story with a paragraph topic in **The Ready Course**, a reading program.**

*Photographs courtesy of SCRIPTAL, University of Michigan
**Photograph courtesy of Central Piedmont Community College

BEST COPY AVAILABLE



FOCUSING ON YOU AND YOUR TEACHING

LET US *"ZOOM IN"* ON YOU. Where are you in your teaching career? What are your goals for your teaching? What are your goals for your students' learning? Do you identify with any of the following statements of advantage mentioned by faculty who have used educational software?

I am able to explain some complex concepts much better.

My students seem to learn from each other more frequently.

I am able to discuss topics that are more challenging.

My students raise more interesting questions in class.

I no longer spend time explaining the same error over and over again.

My students get more tutoring whenever they need it.

I am able to find better ways to motivate my students.

My students spend more time on their courses.

I can get more information about how well my students are doing in my course.

You can turn the above statements into goals.

ADD YOUR OWN GOALS:

Articulating your goals is the first step in applying technology to teaching/learning. The second step is recognizing the assets you bring to the process of using or developing educational software. If you take the time to reflect on your teaching experiences and personal interests, you'll recognize that you are prepared to explore the integration of teaching and technology. Take the following, for example.

If you have observed how different students use your course materials in different ways, you are more likely to evaluate or design educational software to cater to those distinct styles.

If you are skilled at anticipating your students' responses to your questions, you are more likely to include all the possible answers when you design educational software.

If you have challenged your students with "what if" questions, you are more likely to be impressed with the capability of some software to ask and respond to such questions.

If you have conducted research studies on teaching/learning, you are more likely to get excited about the role of technology in implementing the principles of teaching/learning.

If you have used or written programmed textbooks, you are more likely to have some ideas to enhance a tutorial program.

If you enjoy writing or drawing (and do them well!), you are more likely to contribute to the production of educational software and supplementary reading materials for its use.

If you have worked well on group projects, you are more likely to collaborate with others and progress faster in using or developing educational software.

If you have performed well as a group leader, you are more likely to organize a departmental project that focuses on the use or development of educational software.

If you are aware of the support services available in your institution, you are more likely to call upon the appropriate people to help you in the use or development of educational software.

If you enjoy solving complex puzzles that stimulate your thinking, you are more likely to persevere in the development of educational software.

ADD YOUR OWN STRENGTHS (Check with your colleagues who have used or developed educational software.)

What is your next step? You can follow one of two paths:

If you are not interested in developing your own educational software, read Sections III and V.

If you are interested in developing your own educational software, read Sections IV and V.

GETTING INTO THE PICTURE

So you're curious about the next step. Faculty in higher-education institutions, large and small, are using software in varied and unique ways to support their instructional activities. Not all of them developed their own software. Many faculty use existing software in innovative ways to address their concerns about teaching and to enhance their students' learning. There is much to be gained by using existing software to meet your unique teaching needs. For example:

You can use software in your classroom to enhance instruction and to facilitate discussion. The hardware needed to project a computer screen onto a large screen is becoming increasingly affordable.

You can require your students to use computerized (productivity) tools to do their assignments—for instance, by using a word-processing program, spreadsheet, or statistical package to write an essay, prepare a financial report, or analyze data.

You can direct your students to review their lessons and to check their comprehension using computerized tutorials and "drill and practice" modules.

You can challenge your students to develop problem-solving skills through computerized simulations of the real world.

Slowly but steadily, faculty in colleges across the nation have picked up momentum in the use and adaptation of educational software.

How can you get started?

You can use existing educational software as is.

You can adapt existing educational software to meet your needs.

USING EXISTING EDUCATIONAL SOFTWARE

Although this task resembles in some ways the process of selecting a textbook, it is much more time-consuming at present because tested quality software is not available in great abundance. Before you get started, familiarize yourself with what is involved. Starting small is the most sensible strategy.

Scenario 1

As a very small start, you may want your students to have extensive practice in applying the principles they learn in your course to real-life situations. In your conversation with a colleague who teaches the same course, you may find that he/she is using a simulation program to do exactly that. You probe a little further and find that your colleague's students use a specific computer lab on campus where the software is available. You probe even further to determine if your students can use the same lab. Yes, you are told, if you work it out with the lab manager. You meet with the lab manager, who prepares a schedule for your students. You go back to your colleague and determine what kind of instructions (if any) you need to prepare for your students. You are off to a very good start in this scenario, and in less than a week since your first conversation with your colleague, your students may very well be getting the desired practice.

Scenario 2

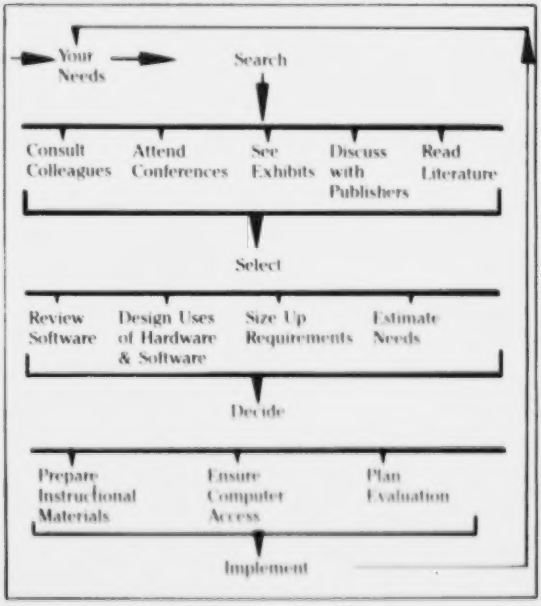
You may want your students to use word processing, spreadsheets, databases, or statistical packages. In this instance, you are dealing with a productivity tool rather than a specific instructional package. Hence you may have to do a little more research. Has your institution settled on one or two productivity tools for institutional use? If there is more than one word-processing program available, which one is familiar to you? Which one is familiar to your students? How much computer-access time do you want each of your students to have? Will the computer labs be able to accommodate this demand? If your institution or department has established procedures for using productivity tools, your task is to find out what they are and follow them. Assuming the computer lab can accommodate

and support your students, it should take only a week or two before your students start working with the productivity tool. On the other hand, if you are the first person who wants the students to use a particular productivity tool, you should expect to do some additional work—procuring the tool for your students' use, becoming familiar with using it, and preparing, if necessary, training materials for your students.

Whether you find yourself in Scenario 1 or 2, you may, after a while, begin to think about expanding your students' use of the software. You may decide to add more lessons, or you may wish to conduct your class in the computer lab once or twice so that you can demonstrate a new program and provide your students with hands-on experience.

START SMALL · DEVELOP EXPERTISE · EXPAND SLOWLY As you expand your uses of existing software, we hope that the following chart will help you stay on track.

THE PROCESS OF USING EXISTING SOFTWARE



The chart is a guide in the process of selecting and using existing educational software. Your first step as a faculty member is to familiarize yourself with the software that is available. You can find such information for your discipline through discussion with colleagues, software demonstrations at conferences, requests to book publishers, and discipline-related publications and newsletters. You'll find a resource list of generic software publishers, journals, and organizations at the end of this booklet.

Among other things, educational software can address any of the following educational needs: simulating the real world for students to explore and learn; providing drill and practice; instructing and training; and/or providing a tool such as word processing to assist students in their work.

Upon selection of software, you should request demonstration copies; most vendors are willing to give a 30-day trial period for educational software. When the software arrives, you should review it for ease of use, adequacy of content, accuracy of instructions, and appropriateness of response (feedback) to students. You can decide to use software either as an additional resource for students or as an enhancement of your instruction in the classroom.

Prior to ordering the software for student use, be sure that the hardware available to students can support the software and the student load; check to see if human support is available to assist the students with the operations of the hardware and software. If the software is to be incorporated into a course, then you will need to prepare the instructional materials and arrange for the hardware and software to be available during your class.

As a final step, you should evaluate the usefulness of the software. This evaluation could include students' reactions, retention rates, grades, and the amount of material covered. Software, like textbooks, will need to be examined yearly to ensure that the content reflects current information. Even though the process of selection, use, and evaluation of software seems unending before you arrive at a sensible and affordable use of it in your setting, the sequence

does become easier as you become more familiar with the available software.

The time and the effort involved in this process may not be adequately acknowledged by the higher-education community. However, as more and more faculty use existing software, more information will become available to explain the process and to suggest the necessary support services.

In all these activities, you don't have to struggle alone. Seek out the librarian, media center director, faculty development director, academic computing coordinator, colleagues in your department, and/or colleagues in other departments for consultation and help.

Seek out your administrative superiors also—department chair and dean. Keep them informed, obtain their endorsement, and seek the necessary funds. If you are enthusiastic and informed about what your project will do for your students, your chances are better that you will get their support. You'll improve your chances even more if you can explain how your project might accomplish your departmental or institutional goals.

ADAPTING EXISTING EDUCATIONAL SOFTWARE There are times when existing software doesn't quite meet your needs. Here are some suggested ways of adapting existing software, based on examples of how some faculty have done it in the past to meet their personal requirements:

- Develop exercises on paper to assist students in effectively using computer-based instruction.
- Use commercially available templates which guide students' use of a particular piece of software.
- Use software with large-screen projection as a demonstration in your class to provide interactive instruction (human and technological), answer "what if's," and perform manipulations.
- Customize software as suggested or allowed by the developer/publisher—for example, establish criterion for passing, alter hardware configurations, or specify content and format for teacher reports.

DEVELOPING YOUR OWN

So you are interested in developing educational software! What is your next step? Perhaps you should examine the opportunities and the responsibilities involved.

START SMALL (ONE PICTURE AT A TIME) Program a little "interaction," a brief tour, or a simple module. Pick something you would really like to do—and have fun doing it! Such an activity will help you see for yourself what is involved in designing and developing a module. Here are some examples to think about:

> Pose a question that the student must answer. Have the program evaluate the answer in such a way that it catches an inappropriate input (such as the wrong letter or a RETURN key), tells the student which keys to use (e.g., "T" or "F"), gives feedback for a right answer and branches the student to another question, and detects a wrong answer and branches the student to an appropriate explanation. You'll see that even this brief interaction can get quite involved. For instance, will you accept lower-case t's and f's?
>
> Display a menu and allow the student to choose an option from among several kinds of review materials and display the appropriate material after the student makes a choice.

Be Aware of the Process of Software Development Whether you plan a large or small project and whether you do it alone or in a group, you should be aware that software development generally involves six stages.

Analysis: setting the objectives and goals of the project; determining the approach; and

ascertaining the hardware and software requirements.

Design: producing a storyboard and flowchart for the program (a visual representation of what the program should do). The table of contents for this booklet is an example of a flow chart.

Programming: coding or otherwise creating the actual software and making sure it conforms to the desired objectives and flowchart.

Documentation: writing a student guide and a faculty manual if the program is expected to be used by other faculty.

Evaluation: testing (debugging) the program both during and after its development to see that it runs smoothly.

Maintenance: providing for updates that will fix bugs and incorporate improvements into the program, if it is expected to be used over a period of time.

SOLO OR TEAM WORK? The production of even a small module of educational software can become complex. Over the years, people with different expertise have learned to work together to develop software, usually using one of at least three models:

Serial Team: The production team consists of a faculty member (a content expert who is also a skilled teacher), an instructional designer, and a computer programmer (the "coder"). In this setting, the teacher starts with an idea, then the designer molds it, and finally the programmer implements it. The teacher then reviews the end product. Some campuses provide support of this kind, and a faculty member wishing to produce educational software may find his or her needs met by this formula.

Collaborative Team: This team is made up of the same individuals as in the serial team, but it works collaboratively from beginning to end. A big advantage in this approach is

the opportunity for collaborative thinking and planning which improves the final product and avoids errors arising out of misinterpretation.

Parallel Team: This team consists of several individual authors, all of whom have teaching experience in the same discipline. Here the programmers and the content experts are one and the same. The efforts at design and development can be shared over time among the authors.

You may for one reason or another prefer to work alone and create whatever captures your imagination. As computers are getting more powerful and complex, numerous tools are becoming available that can assist the individual author. However, before deciding to do it on your own, you should be aware that, in spite of availability of authoring tools, software development can be difficult, and a team approach is highly recommended.

AUTHORING ENVIRONMENT The authoring environment is very critical for smooth and successful development of software. Take your time to check with colleagues who have developed their own software in your institution and in others. Time spent in this activity can help you to minimize your risks and frustrations as you move ahead with developing your own software. One decision to be made early is which authoring language/system to use. You can choose from among the following:

General Purpose Languages (e.g., C, BASIC, Common LISP Pascal): these are programming languages that permit an author an open- ended flexibility to design and develop the software.

Authoring Languages (e.g., Hyperrtalk, PILOT, PODIUM): these languages incorporate shortcuts that coordinate coding with instructional design.

Authoring Systems (e.g., Banque, Camelot, Course of Action,Course Builder, GUIDE, Hypercard, Icon Author, Link Way, PCD3, ProPi, QUEST, TenCore,): these consist of

"no code" interfaces for writing programs. They do not require a knowledge of programming languages, although some of them allow you to write programming commands at the coding level.

As you consider your options for developing educational software, make it your top priority to find a working partner, preferably a programmer, who can complement and support you through the design and development stages.

GETTING READY FOR THE BIG PICTURE Fifty lines of BASIC do not an educational program make! Note well: there is a BIG STEP from short and simple modules to serious software development.

At the rate of sixty to two hundred hours of development time for each hour of instruction in an interactive mode, the production of robust, student-tested, and pedagogically compelling software could require a development period of one to three years or more (including considerable learning curves for hardware and software, etc.).

While skilled use of authoring environments and team efforts can reduce the development time, unless you are merely a "consulting content expert," you should be prepared for a major effort. For example, the EDUCOM/NCRIPTAL (National Center for Research on Improvement in Post Secondary Teaching and Learning) award-winning program, "The Would-Be Gentlemen," was developed over a period of one and one-half years with major institutional and financial support. The work load was as follows:

> Programmer—full time over the summer and fifteen to twenty hours per week during the school year.
>
> Content Expert—fifteen hours per week over the summer and five hours per week during the year.

Before embarking on the development of your own software programs, be sure you understand the following:

Institutional Priorities

What is your institution's commitment in terms of resources, support, and incentives for software development?

Does your college provide support and access to computing?

Will you be able to get release time or special assistance?

Is funding available?

Hardware/Software Choices

How transportable (i.e., able to be used on different brands and models of computers) should your software be?

Will you want your program to be used on different types of computers?

Will any graphics you develop require special hardware?

When new educational needs are identified in the future, can the software be easily expanded or changed to meet those needs?

The Market

See what similar projects have been completed or are under development.

Look at trends in hardware and operating systems and identify which will be used in the future.

Avoid conditions that cause programs to become outdated or that require an esoteric environment.

Investigate the available publishing and distribution channels. Identify software publishers that currently have produced or have announced plans to produce and/or distribute higher-education software. Book publishers are generally interested in programs that wrap around particular textbooks.

Check if your institution is interested and has the capability to market the software you plan to produce.

Look into software distribution channels such as WISC-WARE, Kinko's Academic Software Exchange, and CONDUIT. (See the Resource List for addresses.)

Be sure to research legal aspects such as copyright and licensing arrangements.

OUT OF THE DARKROOM Developing your own software module on a small scale can give you important insights into the process of educational software development. You may also choose to venture further into the challenging arena of software design and development for an entire course. Because this section has attempted to give you a wide-angle view of both activities, it has not given you all the details on every aspect of educational software development. If it has opened your eyes to the opportunities and responsibilities, then it has accomplished its goal.

ENLARGING THE PAY-OFFS

Many faculty who have used or developed educational software speak of the "value-added" effect of their effort. These values, both intrinsic and extrinsic are listed below:

ENLARGES ONE'S UNDERSTANDING OF TEACHING, LEARNING, AND TECHNOLOGY:

Prompts one to see the "holes" in the treatment of a topic.

Increases one's sensitivity to students' difficulties in learning.

Helps one to recognize the strengths and limitations of technology.

Deepens one's understanding of the subject matter.

Stimulates one to rethink what to teach and how to teach.

ENHANCES STUDENT-FACULTY RELATIONSHIPS:

Makes a systematic analysis of student performance possible.

Provides accurate student data to facilitate faculty action.

Permits more time for faculty-student and small group discussions.

Saves time from "drill and practice" activities.

Allows students to get the help they need when they need it.

PROVIDES OPPORTUNITIES FOR PROFESSIONAL GROWTH:

Expands one's domains of knowledge in the discipline.

Increases one's skills in evaluating existing material.

Heightens one's sense of professionalism.

Stimulates one's excitement about teaching.

Strengthens one's reputation as a good teacher.

Increases one's chances for publishing and making conference presentations.

AUGMENTS THE INSTITUTION'S PRESTIGE:

Increases the visibility of the institution.

Helps the institution to obtain external grants.

What would you consider as your pay-off?

WHERE ARE YOU IN THE PICTURE?

RESOURCE LIST:

AMERICAN ASSOCIATION OF COMMUNITY AND JUNIOR COLLEGES (AACJC)
1 Dupont Circle, Suite 410
Washington, DC 20036
(202) 293-7050

ASSOCIATION FOR THE DEVELOPMENT OF COMPUTER-BASED INSTRUCTIONAL SYSTEMS (ADCIS)
Miller Hall 409
Western Washington University
Bellingham, WA 98225
(206) 676-2860

ASSOCIATION FOR EDUCATIONAL COMMUNICATIONS & TECHNOLOGY
1126 Sixteenth St., N. W.
Washington, DC 20036
(202) 466-4780

EDUCOM
P.O. Box 364
777 Alexander Road
Princeton, NJ 08540
(609) 520-3343

LEAGUE FOR INNOVATION IN THE COMMUNITY COLLEGE
23276 South Point Drive, Suite 103
Laguna Hills, CA 92653
(714) 855-0710

NCRIPTAL
2400 School of Education Building
The University of Michigan
Ann Arbor, MI 48109
(313) 936-2743

NON-COMMERCIAL SOFTWARE DISTRIBUTORS:

CLEARINGHOUSE FOR ACADEMIC SOFTWARE COMPUTATION CENTER
104 Computer Science Building
Iowa State University
Ames, IA 50011
(515) 294-0323

CONDUIT
University of Iowa
Oakdale
Iowa City, IA 52242
(319) 353-5789

KINKO'S ACADEMIC COURSEWARE EXCHANGE
4141 State Street
Santa Barbara, CA 93110
(800) 235-6919

WISC-WARE
Academic Computing Center
University of Wisconsin
210 West Dayton Street
Madison, WI 53706
(608) 262-8167

JOURNALS AND BULLETIN BOARDS:

ACADEMIC COMPUTING
P.O. Box 532
McKinney, TX 75069
(214) 548-2101

COLLEGIATE MICROCOMPUTER
Rose-Hulman Institute of Technology
Terre Haute, IN 47803

THE ESI NEWSLETTER
EDUCOM Software Initiative
PO. Box 364
Princeton, NJ 08540
(609) 520-3340

IBM INFOWINDOW NEWSLETTER
IBM Advanced Education Systems
PO. Box 2150
Atlanta, GA 30055
(404) 238-4646

INNOVATOR
League for Innovation in the Community College
25431 Cabot Road, Suite 203
Laguna Hills, CA 92653
(714) 855-0710

INSTRUCTION DELIVERY SYSTEM
50 Culpepper Street
Warrenton, VA 22186
(703) 347-0055

ISAAC: The Information System for Advanced Academic Computing
M/S FC-06
University of Washington
Seattle, WA 98195

JOURNAL OF COMPUTER-BASED INSTRUTION
Association for the Development of Computer-Based Instructional Systems
Miller Hall 409
Western Washington University
Bellingham, WA 98225
(206) 676-2860

T.H.E. JOURNAL
(Technological Horizons in Education Journal)
Information Synergy, Inc.
Box 17239
Irvine, CA 92713
(714) 261-0366

TELETRENDS
Cyahoga Community College
700 Carnegie Avenue
Cleveland, OH 44115
(216) 987-3026

WHEELS FOR THE MIND
Apple Computer, Inc.
Wheels for the Mind - 23L
PO. Box 810
Cupertino, CA 95015

A JOINT PUBLICATION OF

EDUCOM

© 1989, EDUCOM ®. All rights reserved.

Interuniversity Communications Council, Inc.
EDUCOM, PO Box 364, Princeton, NJ 08540
1112 16th Street NW, Washington, DC 20036

ERIC Clearinghouse for
Junior Colleges
DEC 08 1989